Jan - March 1998

102(b).

ProQuest

Help

Basic Search | Advanced Search | Topic Guide | Publication Search | Marked List :0 article | Language: English

Databases selected: Multiple databases...

Article View « Back to Results Article 1 Publisher Information

Print Email Mark Article Abstract , Full Text , Text+Graphics , Page Image - PDF

SpeedTracer: A Web usage mining and analysis tool

K-L Wu, P S Yu, A Ballman. IBM Systems Journal. Armonk: 1998. Vol. 37, Iss. 1; pg. 89, 17 pgs

» **Jump to full text**

Subjects:	Product design, Algorithms, World Wide Web, Data mining, Computer programming
Classification Codes	7500 Product planning & development, 5240 Software & systems
Author(s):	K-L Wu, P S Yu, A Ballman
Publication title:	IBM Systems Journal. Armonk: 1998. Vol. 37, Iss. 1; pg. 89, 17 pgs
Source Type:	Periodical
ISSN/ISBN:	00188670
ProQuest document ID:	26217513
Text Word Count	7845
Article URL:	http://gateway.proquest.com/openurl?ctx_ver=z39.88-2003&res_id=xri:pqd&rft_val_fmt=ori:fmt:kev:mtx:journal&genre=article&rft_id=xri:pqd:did=000000026217513

More Like This »Show Options for finding similar articles

Abstract (Article Summary)

SpeedTracer, a World Wide Web usage mining and analysis tool, was developed to understand user surfing behavior by exploring the Web server log files with data mining techniques. With innovative algorithms, **SpeedTracer** first identifies user sessions by reconstructing user traversal paths. It does not require "cookies" or user registration for session identification. User privacy is protected. Once user sessions are identified, data mining algorithms are then applied to discover the most common traversal paths and groups of pages frequently visited together. Three types of reports are prepared: user-based reports, path-based reports, and group-based reports. The design of **SpeedTracer** is described and some of its features are demonstrated with a few sample reports.

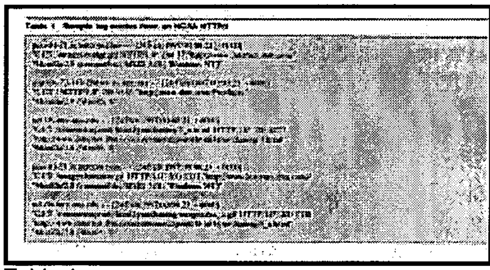
Full Text (7845 words)

SpeedTracer, a World Wide Web usage mining and analysis tool, was developed to understand user surfing behavior by exploring the Web server log files with data mining techniques. As the popularity of the Web has exploded, there is a strong desire to understand user surfing behavior. However, it is difficult to perform user--oriented data mining and analysis directly on the server log files because they tend to be ambiguous and incomplete. With innovative algorithms, **SpeedTracer** first identifies user sessions by reconstructing user traversal paths. It does not require "cookies" or user registration for session identification. User privacy is protected. Once user sessions are identified, data mining algorithms are then applied to discover the most common traversal paths and groups of pages frequently visited together. Important user browsing patterns are manifested through the frequent traversal paths and page groups, helping the understanding of user surfing behavior. Three types of reports are prepared: user-based reports, path-based reports and group-based reports. In this paper, we describe the design of **SpeedTracer** and demonstrate some of its features with a few sample reports.

Popularity of the World Wide Web (www) on the Internet has exploded recently. Many organizations have invested a tremendous amount of capital to operate sites on the Web. These Web sites provide communications and services to their employees, customers, and suppliers. With money invested in these sites, there is a strong desire to

understand the effectiveness of such investments and to find ways to realize the potential opportunities provided by the Internet. As a result, it has become important to understand user surfing behavior.

To understand how visitors navigate a Web site, the Web server log files are analyzed. However, it is generally difficult to perform user-oriented data mining or analysis directly on the server log files because they tend to be ambiguous and incomplete. Typical server log files contain the following information about a request: client host Internet Protocol (IP) address, time stamp, method, URL1 (uniform resource locator) address of the requested document, HT[PZ (HyperText Transfer Protocol) version, return code (status of the request, i.e., success or error codes), bytes transferred, referrer page URL, and agent (browser and client operating system). The user identifier is usually not available in the log file. Due to the use of proxy servers by Internet Service Providers (ISPs) and firewalls by commercial corporate gateways, true client IP addresses are not available to the Web server. Instead of various distinct client IPs, the same proxy server or firewall IP will be recorded in the server log files, representing requests of different users who come to the Web site through the same proxy server or firewall. This situation creates ambiguity in the log records. Furthermore, some Web pages are generally cached by local clients or various proxy servers, or both, in order to reduce network traffic. As a result, log records will be missing for the corresponding accesses to the cached Web pages, resulting in an incomplete log. A more complete discussion of the difficulties in obtaining reliable usage data on the Web can be found in Reference 3.



Enlarge 200%
Enlarge 400%

Table 1

For example, Table 1 shows a few sample entries of an access log in the combined log format from a National Center for Supercomputing Applications (NCSA)4 HTTPd.5 The first entry in Table 1 represents a GET request from a user going through peo-ill21.ix.netcom.com for file /images/nudge.gif following HTTP/1.0 protocol. The user may or may not be physically logged in on the machine peo-ill-21.ix.netcom.com. He or she may be just using the machine as a gateway to the Internet. The file size of nudge.gif is 37 bytes, and it was successfully transferred. The agent used to view page nudge.gif is MSIE** 3.01 (Microsoft Internet Explorer** 3.01) running on Windows NT**. Finally, the user was referred to the "gif, file from http: //www.internet.ibm.com/. Namely, either file nudge.gif is on the home page of http://www.internet.ibm.com/ or there is a hyperlink to it from the home page.

To solve the problem of proxy servers or firewalls masking user IPs, it generally requires either user registrations or log-ins or the employment of "cookies" between the Web server and client browsers. With log-ins or cookies, a Web server can identify distinct requests made by individual users through a token carried between the user's browser and the server.

But the desire by many, if not the majority of, users to have privacy and remain as anonymous as possible may force many Web servers not to ask for registration or not to use cookies. As a result, there is a strong need for a tool that can analyze user-oriented behavior from the regular server log files without requiring cookies or registrations. **SpeedTracer**,6 a Web usage mining and analysis tool, has been developed for such a purpose.

Several Web server log analysis tools have been implemented. Some of these tools are very simple and do not attempt to identify individual user sessions. These packages are simply mechanisms through which a Web master can view the raw Web server statistics, such as hit counts and distributions based on geographic regions. Examples of this type of tool include wwwstat (http://www.ics.uci.edu/pub/websoft/wwwstat) and Analog (http://www.statslab.cam.ac. uk/~sret1/analog).

To provide user-oriented Web usage analysis, user sessions must first be identified. More sophisticated analysis packages identify user sessions with some or all of the following three mechanisms. First, if the Web server provides cookies, it is a trivial task to formulate the session. Every access to the Web server with the same cookie value makes a single session. Second, if the server does not provide cookies, it may require a log-in ID for each browser. The analysis tool can use the log-in IDs to identify sessions. In Reference 7, a data mining tool was developed on the assumption that log-in IDs are available. Without log-in IDs, the data mining tool cannot perform its intended

functions. In fact, most log records do not contain log-in IDs. Lastly, if the Web server does not provide either cookies or user IDs, the analyzer identifies sessions with host addresses. All accesses to the Web server from a given host address are considered to be a session until a predefined amount of time has passed between accesses. As mentioned previously, the use of a proxy and firewall causes all browsers from a given proxy or firewall to be considered a single user. As a result, an identified session may in fact contain many independent user sessions. Several Web analyzers only use the host address to identify sessions, such as SurfReport** (http://software.bienlogic.com/SurfReport) and NetTracker** (http://www.sane.com/products/ NetTracker). Other tools use a combination of methods to identify sessions, such as the Usage Analyst** by Microsoft Corporation (previously Interse http://www.interse.com) and WebTrends** (http: //www. webtrends.com).

In contrast, **SpeedTracer** uses the referrer page and the URL of the requested page as a traversal step and reconstructs the user traversal paths for session identification. No "cookies" or user registration are required. In Reference 8, an alternative approach to reconstructing user traversal paths was proposed. Instead of using a referrer page, information about the topology (i.e., hyperlink structure) of a Web site (together with other heuristics) was used to identify legitimate traversals. A software agent was first used to perform an exhaustive breadth-first traversal of pages within the Web site in order to construct the topology. However, the topology is not really needed if referrer information is available.

Stats

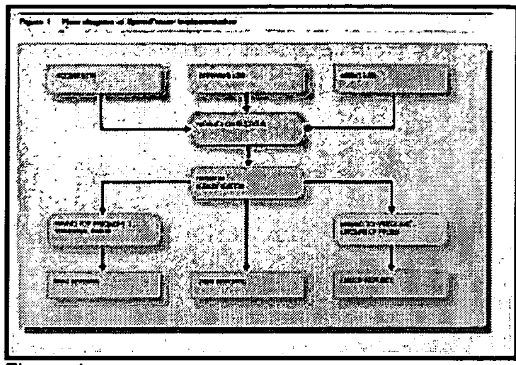
Once user sessions are identified, statistics related to user behavior can be obtained. Interesting user-- based statistics include the top N referrers to a Web site, the top N pages most frequently visited by users, the top N pages from or into which users most frequently exit or enter a Web site, the top N browsers most frequently used, the top N IP hosts from which most users come, the demographics (by organization or by country) of users, the distribution of user session durations, the distribution of numbers of pages visited during a user session, and the distribution of depth or breadth of a user session.

With user sessions, data mining techniques can be applied to obtain interesting user browsing patterns. Data mining has recently been used to discover customer buying patterns by many retailers and other service corporations. One of the most important data mining problems concerns mining association rules.9-12 Given a set of transactions, where each transaction is a set of items, an association rule is an expression of the form X -> Y, where X and Y are sets of items. An example of an association rule is: "30 percent of transactions that contain bread and butter also contain milk; 2 percent of all transactions contain both of these items." Here 30 percent is called the confidence of the rule, and 2 percent the support of the rule. The thrust of mining association rules is to find all association rules that satisfy user-specified minimum support and minimum confidence constraints. In mining association rules, the most important problem is to generate all combinations of items that have the minimal support. These combinations of items are called large itemsets.

In **SpeedTracer**, we mapped each identified user session into a transaction and then applied data mining techniques to discover the top N most frequented user traversal paths and the top N groups of pages most frequently visited together. These problems are to some extent similar to finding the top N large itemsets for traversal paths and groups of pages. But specific differences exist. A traversal path is a collection of consecutive URL pages in a Web presentation, where one URL is referred to by the immediately preceding URL. The URLs in a traversal path are connected in the Web presentation graph. In contrast, the pages in a group are not necessarily connected among themselves. A frequently visited group of pages may contain two or more disjoint traversal paths. By examining the traversal paths and groups of pages, valuable user browsing patterns can be obtained to improve the organization and linkage of the Web presentation.

Note that finding frequent traversal paths is also to some extent similar to the problem of mining sequential patterns.13 However, the results from the sequential-pattern-mining method in Reference 13 may contain sequences that do not represent a traversal path in the Web presentation graph. The reason is there may be many backward traversal steps involved in a user session, and pages on two different paths may be recognized as part of a sequential pattern. In this paper we focus on mining the frequent traversal paths and groups of pages visited together. However, other types of data mining techniques, such as clustering pages, clustering users, or classification, could be applied to the derived user sessions. These and other techniques will be explored in future work.

The next section describes the design and implementation of **SpeedTracer**. The implementation details of session identification, mining of frequent traversal paths, and mining of groups of pages most frequently visited together are presented. The section following the next one shows a few sample reports from **SpeedTracer**. User reports, path reports, and group reports are highlighted. Finally, we conclude with a summary.



Enlarge 200%
Enlarge 400%

Figure 1

Design of **SpeedTracer**

In **SpeedTracer**, we first process the access, referrer, and agent logs (or just access log, if it is in combined log format) to identify user sessions. If referrer and agent information are stored in separate files, delicate synchronization procedures may be needed to relate an access log record to the corresponding referrer and agent records because log entries may be missing in some of the files. For example, referrer or agent log records, or both, may be missing from NCSA logs. However, in IBM's ICS (Internet Connection Server) logs, every access log record has corresponding referrer and agent log records even if they are stored in separate files. **SpeedTracer** is designed to process different kinds of log formats, such as the NCSA log, the IBM ICS log, and others. For simplicity, in this paper we use the IBM ICS log as an example. Figure 1 shows the flow diagram of **SpeedTracer** implementation. After log records are processed, user sessions are identified using advanced inference algorithms. Interesting user-based statistics are then computed based on these sessions. Next we apply data mining techniques to discover the top N most frequented user traversal paths. Finally, we discover the top N groups of pages most frequently visited together.

User session identification. Given the information available in the server log, a possible approach to grouping various accesses into user sessions is to use both time stamps and agent information. For example, a user session could include all accesses within a predetermined interval if their agents are the same. Unfortunately, this approach cannot distinguish two different clients with the same agent coming from the same proxy within the specified time interval. For instance, in Table 1 the two accesses from ss5-08. inre.asu.edu both use Netscape Navigator** 3.0 (Mozilla/3.0) and come within four seconds. The two accesses can be viewed as from the same user session. But they may be from two different user sessions going through the same proxy server. As the markets for both browsers and desktop operating systems become ever more consolidated, it is highly likely that multiple accesses from different users will have the same agent. For instance, most home users may use the same version of Netscape Navigator browser running on a Windows 95** desktop. Thus, time stamps together with agent information are not sufficient to identify user sessions from the server log files.

In **SpeedTracer** we use five key pieces of information from a log record to identify user sessions. They are IP, Timestamp, URL (the requested page), Referral, and Agent. Different IPs or agents obviously indicate different user sessions. If the time stamps indicate that two accesses are separated by more than a prespecified period of time, the accesses are also considered to belong to different sessions. In addition to these obvious rules, **SpeedTracer** uses the referral page to help more accurately identify user sessions. For each log record, we use the referral page and the requested page URL to form a hyperlink access pair, representing a step in a user traversal path. Each access pair is then used to reconstruct a user traversal path in the Web presentation. The basic idea is that access pairs constitute a connected traversal path during a user session. Note that the traversal path can be forward or backward. Session identification becomes the partitioning of log records into groups so that the access pairs within a group form a connected traversal path. However, because browsers and proxy servers generally use caching to reduce network traffic and improve performance, there are no corresponding log entries for those accesses to the cached pages. As a result, missing access pairs might be in the log files, and these missing access pairs need to be added back during session identification.

In session identification, we process the log records one at a time. Each access pair is added to an active session, if possible. If (x^sub i ^ -> y^sub i^) represents an access pair, then the traversal path of a session S of size n can be expressed as follows:

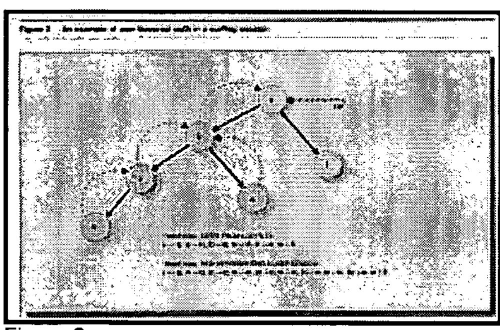
S: (x^sub 1^ -> y^sub 1^), (x^sub 2^ -> y^sub 2^), . . . , (x^sub n^ -> y^sub n^)

where x^sub i + 1^, = y^sub i^, 1 <= i < n. A new access pair (x^sub j^ -> y^sub j^ can be appended to an active session S, if x^sub j^ = y^sub k^, 1 <= k <= n, or x^sub 1^ = x^sub j^. However, unless x^sub j^ = y^sub n^, a backward access path (y^sub n^ -> x^sub n^), . . . , (y^sub k = 1^ -> x^sub k + 1^) must first be added back to S to maintain a connected traversal path. For example, if (b -> d) were to be appended to a session S^sub i^: (a -> b), (b -> c), a backward traversal pair (c -> b) has to be first appended to S^sub i^. Thus, the new S^sub i^ becomes (a -> b), (b -> c), (c -> b), (b -> d).

Apparently, there can be multiple candidate sessions to which a new access pair can be appended. Different criteria or combinations of them can be used to choose one candidate session. For example, one criterion can be the number of backward access pairs needed to be added. Another criterion can be the time-stamp difference between the access pair and a session. The time stamp of a session is the time stamp of its latest appended access pair. Combinations of these two criteria can also be used. For example, one can choose the session with the smallest time-stamp difference with backward access pairs no more than m, or the one with the smallest number of backward access pairs with time-stamp difference no more than q minutes. Advanced inference algorithms are developed for this purpose.

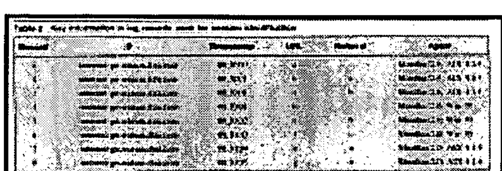
As an example, Table 2 shows the key information of eight example log records used by **SpeedTracer** for session identification. These eight log records represent requests coming from a gateway for the IBM Watson Research Center. From Table 2, the hyperlink access pairs for the eight log records are (- -> a), (e -> b), (b -> c), (- -> b), (b -> c), (- -> f), (a -> b), (b -> g), respectively. Here, "-" means that no referral page is available for this access. Using these access pairs and the agent information, we can identify four user sessions as follows: S^sub 1^: (- -> a), (a -> b), (b-> g) from log records 1, 7, and 8; S^sub 2^: (e -> b), (b -> c) from log records 2 and 3; S^sub 3^: (- -> b), (b -> c) from log records 4 and 5; and S^sub4^: (- -> f) from log record 6. Note that if we were to use only time stamp and agent for session identification, we would have grouped log records 1, 2, 3, 7, and 8 as a user session and log records 4, 5, and 6 as another user session. However, from the referral information of both log records 1 and 2, it is obvious that these two are from different user sessions. The access to page b in log record 2 must be following a previous access to page e, whereas the access to page a may be the beginning of a new user session since it did not contain a referral.

The need for inferring backward access pairs in session identification due to proxy or client caching can be illustrated with the following example. Figure 2 shows a traversal pattern by a Web user during a surfing session. Note that any access pair (x -> y) can be the result of clicking a hyperlink to page y on page x or by clicking on the backward button on the Web browser to page y after the viewer has looked at page x. The user session represented by the connected traversal path in Figure 2 can be described as follows: (- -> a), (a -> b), (b -> c), (c -> d), (d -> c), (c -> b), (b -> e), (e -> b), (b -> a), and (a -> f). However, since browsers usually cache recently visited pages, some of the actual traversal steps may be missing from the server log files. For example, in Figure 2, (d -> c), (c -> b), (e -> b), and (b -> a) may be missing. These missing traversal steps may need to be inferred in order to identify traversal paths and user sessions.



Enlarge 200%
Enlarge 400%

Figure 2



Enlarge 200%
Enlarge 400%

Table 2

Since a "gif" or "jpg" file typically does not expand a traversal path, we eliminate all log records whose URL contains

these graphical files in our session identification. **SpeedTracer** also takes care of special cases caused by a user's clicking on the "reload" button and "bookmarking" his or her hot links. On a reload, the repeated access pair is discarded since it does not expand a traversal path. If an access is the result of a user accessing the Web page through his or her bookmark or directly typing in the URL, no referrer information is available on the log record. In **SpeedTracer**, we view this as the beginning of a new session. Once sessions are identified, user-oriented statistics can be obtained.

Interesting user-based statistics are provided by **SpeedTracer**, including the most frequent N external referrers to a site, the most frequent N visited pages by users, the most frequent N pages that users most often come into and exit from a site, the top N hosts from which most users come to visit a site, the distribution of user session durations, and the number of pages visited in a session. Sample reports and their applications will be presented in the next section. N can be specified by a user of **SpeedTracer** before it analyzes the log files and prepares the reports.

Mining frequent traversal paths. Once user sessions are identified, the problem of mining frequent traversal paths becomes a matter of discovering the most frequent subpaths common among all the sessions. In finding user traversal patterns, we are only interested in forward traversal subpaths. As a result, **SpeedTracer** first finds all maximum forward paths in each user session, and then discovers all common subpaths among all the maximum forward paths of user sessions. A maximum forward path is a sequence of maximum connected pages in a Web presentation where no page is previously visited. 14 For example, in Figure 2, three maximum forward paths are in this session: (1) (a -> b) (b -> c) (c -> d); (2) (a -> b) (b -> e); and (3) (a -> f). Figure 3 shows the algorithm for finding all maximum forward paths from a user session. Assume {x^sub 1, . . . , x^sub m^} represents a user session and {y^sub 1, . . . , y^sub j-1^} represents a string holding a potential maximum forward path. The idea is to examine each page xi in the session one at a time and try to expand the potential maximum forward path by copying x^sub i^ to y^sub j^, if x^sub i^ is not equal to any y^sub k^ for every 1 <= k < j. Namely, the pages in the potential maximum forward path are all distinct pages, and we are going in the forward direction in the user traversal path. We use a flag to indicate that we are currently moving in the forward direction in path traversal. In contrast, if x^sub i^ is equal to some y^sub k^, 1 <= k < j, then we are going backward in the traversal path, and subpath {y^sub 1, . . . , y^sub j-1} can be a maximum forward path if the flag indicates that we have been going in the forward direction before this step. After discovering matched page y^sub k^, we eliminate pages {y^sub k = 1^, . . . , y^sub j-1^} from the potential maximum forward path by moving j backward to k + 1 for the next iteration, and set the flag to indicate backward direction. At the end, if the flag indicates forward direction, the final subpath is the final maximum forward path for the session.

As an example, Table 3 shows the values of subpath {y^sub 1^, . . . , y^sub j-1^} and the flag at the end of each execution step of finding maximum forward traversal paths. We use the user session in Figure 2 as our input. If we represent the session as a sequence of pages visited, this session is {a, b, c, d, c, b, e, b, a, f}, and the three maximum forward paths identified are {a, b, c, d}, {a, b, e}, and {a, f}. For the first four steps, we are going in the forward direction and expanding the potential maximum forward path. In step 5, page c is found in subpath {a, b, c, d}, so a maximum forward path is found, and the traversing direction is reversed. Such a reversal lasts for two steps until step 7 when page e is expanded again to form {a, b, e}. In step 8, page b forces {a, b, e} out as another maximum forward path and reverses the traversal direction. At the end, {a, f} is found as a maximum forward path since the flag indicates forward direction.

Once the maximum forward paths are constructed for each session, we then map the problem of finding the top N frequent traversal paths into the one of finding frequently occurring consecutive subsequences among the maximal forward paths of all user sessions. A large traversal path is a sequence of consecutive pages that appeared in the maximal forward paths of a sufficient number of sessions. The number of sessions in which a large traversal path appears is called its support. A large traversal path of size k contains k pages. In this paper, we denote the set of top M large traversal paths of size k as LP^sub k^.

Note that a significant difference exists between discovering large itemsets in mining association rules and discovering large traversal paths in mining traversal patterns. In a large traversal path, the pages must form a consecutive sequence in a maximal forward path, whereas a large itemset in mining association rules is just a set of items in a transaction.

Assume that P^sub k,M^ is the Mth largest traversal path in LP^sub k^ (the support of P^sub k,M^ is the Mth largest); s^sub k^ is the support of Pk,M; Fi is the set of maximum forward paths for session S^sub i^; and {x^sub 1^, x^sub 2^, . . . , x^sub m} is the sequence of pages representing a maximum forward path in F^sub i^. The algorithm for constructing LP^sub k^, k > 1 can be described as in Figure 4. After each LP^sub k^ is constructed, the top N traversal paths are then reported. In **SpeedTracer**, we set M to be greater than N in computing each LP^sub

k^. Namely, we computed more large traversal paths for each LP^sub k^ than we reported. LP^sub 1^, is the set of all single pages, and s^sub 1^ is the number of user sessions that referenced the Mth hottest page.

Figure 3 Algorithm for finding all maximum forward paths from a session

```
y1 = x1; i = 2; j = 2; flag = YES;
while (i ≤ m) {
   if (xi == yk) for some 1 ≤ k < j {
     if (flag == YES)
       output (y1, ..., yj-1) as a maximum forward path;
     j = k + 1; i = i + 1; flag = NO;
   }
   else {
     yj = xi; j = j + 1; i = i + 1;
     flag = YES;
   }
}
if (flag == YES) {
   output (y1, ..., yj-1) as the final maximum forward path;
```

Enlarge 200%
Enlarge 400%

Figure 3

Figure 4 Algorithm for discovering large traversal path set LP_k

```
for each Fi {
   for each {x1, x2, ..., xm} in Fi {
     if (m ≥ k) {
       for (j = 1; j ≤ m - k + 1; j++) {
         if ({xj, ..., xj+k-1} is already in LPk)
           increase its corresponding count;
         else if ((support of {xj, ..., xj+k-2} ≥ sk-1) and
                 (support of {xj+1, ..., xj+k-1} ≥ sk-1))
           insert {xj, ..., xj+k-1} into LPk;
       }
     }
   }
}
```

Enlarge 200%
Enlarge 400%

Figure 4

Table 3 Example execution steps of finding maximum forward paths

Step	x_i	Subpath $\{y_1, \ldots, y_{j-1}\}$	Flag	Maximum forward path
1	a	$\{a\}$	YES	
2	b	$\{a, b\}$	YES	
3	c	$\{a, b, c\}$	YES	
4	d	$\{a, b, c, d\}$	YES	
5	c	$\{a, b, c, d\}$	NO	$\{a, b, c, d\}$
6	b	$\{a, b\}$	NO	
7	e	$\{a, b, e\}$	YES	
8	b	$\{a, b\}$	NO	$\{a, b, e\}$
9	a	$\{a\}$	NO	
10	f	$\{a, f\}$	YES	
11		$\{a, f\}$	YES	$\{a, f\}$

Enlarge 200%
Enlarge 400%

Table 3

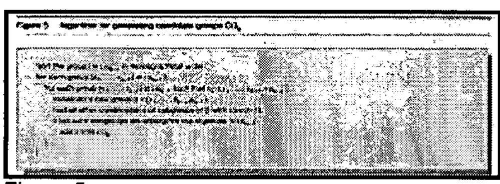
The idea of constructing LP^sub k^ is to find a candidate path of size k, {x^sub j^, . . . , x^sub j=k-1}, from a maximum forward path and then compute its occurrence count among the maximum forward paths of all user sessions. The condition is that the two consecutive subsequences of size k - 1, {x^sub j^, . . . , x^sub j+k-2} and {x^sub j=1, . . . , x^sub j+k-1} are among the top M largest traversal paths in LP^sub k-1^. For example, assume session S^sub 1^, contains two maximum forward paths: {A, B, C, D, E} and {G, H}. To consider candidate large paths of size 3 for inclusion in LP^sub 3^, we would test three candidate large paths: {A, B, C}, {B, C, D}, and {C, D,

E}. If both {A, B} and {B, C} are among the top M large paths in LP^sub 2^, then {A, B, C} is a candidate for LP^sub 3^. Similarly, if both {B, C} and {C, D} are in LP^sub 2^, then {B, C, D} can be included in LP^sub 3^.

In Figure 4, for each candidate large path of size k, {x^sub j, . . . , x^sub j+k-1^} from the maximum forward paths of a user session, we increment its occurrence count if it already is in LP^sub k^. There are (m - k + 1) total consecutive subsequences of size k from {x^sub 1, x^sub 2^, . . . , x^sub m^} . For example, there are three such candidate consecutive subsequences of size 3 ({A, B, C}, {B, C, D}, and {C, D, E}) from a maximum forward path of size 5 ({A, B, C, D, E}) as we just illustrated above. We examine each one of them. If the subsequence of size k has not already been included in LP^sub k^, we test to see if it can be. If yes, we add {x^sub j^, . . . , x^j+k-1^}; otherwise we do nothing. The conditions here are based on the fact that if a traversal path of size k is among the top M largest in LP^sub k^, then its two consecutive subsequences of size k - 1 must be also among the top M largest in LP^sub k-1^ . Obviously, if m < k, nothing needs to be done for this maximum forward path. For instance, nothing needs to be done for {G, H} for LP^sub 3^. Note that for each k, all the maximum forward paths of all user sessions are scanned only once.

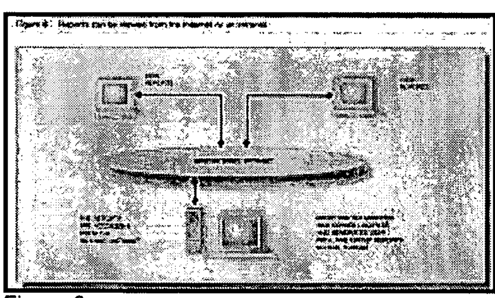
Mining groups of pages most frequently visited. Frequent traversal paths identify pages that are on the same forward path in a Web presentation. These pages represent consecutive subsequences in the maximum forward paths of user sessions. However, there might be groups of pages not on the same traversal path but frequently visited together by users. By examining both frequented traversal paths and frequently visited groups, valuable information can be obtained to improve the organization and linkage of the Web presentation. For example, in Figure 2, pages a, b, e, and f may be visited most frequently by users, but these four pages are not on the same path in the Web presentation. Thus, it may be better to provide an HTTP link from page e to page f so that most users would not have to traverse backwards from page e to b, then to page a before they can go to page f.

To mine the frequently visited groups from user sessions, we need the distinct pages in each session. Thus, any duplication of pages caused by backward traversals was first eliminated in each session. Unlike traversal paths where the ordering of pages in a sequence is important, there is no ordering in a group of pages. Similar to mining the traversal paths described above, let us assume that LG^sub k^ is the set of top M largest groups, each consisting of k pages, and the support of the smallest group in LG^sub k^ is Sk. Unlike mining traversal paths, however, L G^sub k+1^, cannot be efficiently constructed directly because of the numerous possible combinations of candidate groups of size k + 1 from each session. For example, a maximum forward path {x^sub 1^, . . . , x^sub m^} has (m - k + 1) candidate traversal paths of size k. But a session {x^sub 1^, . . . , x^sub m^} will have C^sup m^^sub k^ (or m!/k!(m - k)!) candidate groups of size k. As a result, **SpeedTracer** constructed LG^sub k+1^ by (1) generating a set of candidate groups of size k + 1, denoted as CG^sub k+1^, from LG^sub k^, and (2) counting the occurrences of each group in CG^sub k+1^ against all sessions. The approach to mining frequently visited groups of pages is thus similar to the discovery of large itemsets in most prior literature of mining association rules.9,10,12



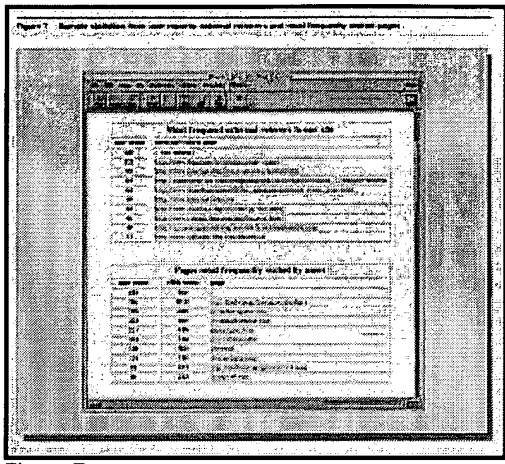
Enlarge 200%
Enlarge 400%

Figure 5



Enlarge 200%
Enlarge 400%

Figure 6

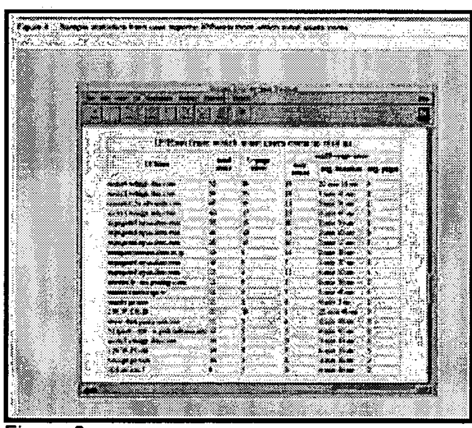


Enlarge 200%
Enlarge 400%

Figure 7

Similar to LP_{1}, LG_{1} contains the top M hottest pages referenced by user sessions. As pointed out in Reference 12, because of the nature of C^{M}_{2}, the computations of CG_{2} and LG_{2} can be substantially more demanding than other CG_{k} and LG_{k} fork > 2. This is in contrast to the case of mining traversal paths, where the candidate paths of LP_{2} cannot be more than the number of links in the Web presentation. Therefore, special treatment is needed. 12

The task of generating CG_{k} (candidate groups set) from LG_{k-1} can be described as in Figure 5. Note that we generate CG_{k} from LG_{k-1}. To simplifyenumerating all possible combinations of groups, we sort the M groups in LG_{k-1}, based on their lexicographical order. The basic idea here is to find all possible groups of size k from LG_{k-1}, based on the fact that for a group of size k to be a candidate group in CG_{k}, all its subgroups of size k -1 must be in L G_{k-1}. 9,10 So, we first try to construct a group of size k for each $\{x_{1}, \ldots, x_{k-1}\}$ in LG_{k-1} by finding all the $\{y_{1}, \ldots, y_{k-1}\}$ in LG_{k-1} such that $x_{2} = y_{1}, \ldots, x_{k-1} = y_{k-2}$. The new k-sized group is thus an expansion of $\{x_{1}, \ldots, x_{k-1}\}$ with $\{y_{k-1}\}$. In order for such new k-sized group to be included into CG_{k}, all the combinations of (k - 1)-sized subgroups of it must all be in LG_{k-1}.



Enlarge 200%
Enlarge 400%

Figure 8

Enlarge 200%
Enlarge 400%

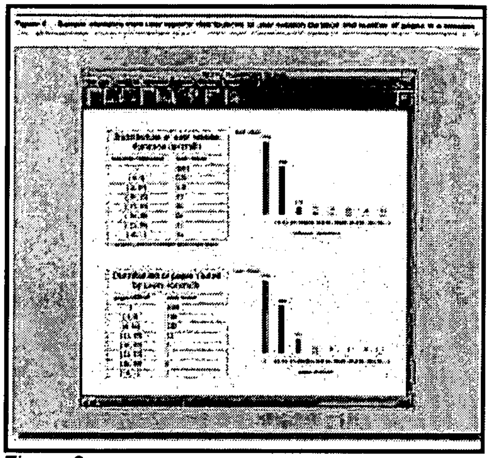


Figure 9

Once CG^sub k^ is generated, we scan all the user sessions one at a time and increment the occurrence count of each candidate group in CG^sub k^ if a session contains all the pages in the group. Upon completion, the top M candidate groups in CG^sub k^ become LG^sub k^.

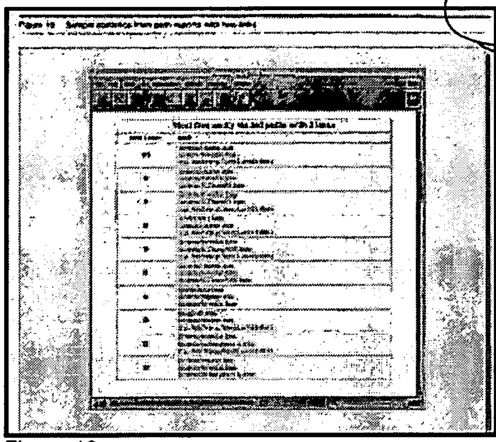
Sample reports from **SpeedTracer**

Now that we have described the implementation of **SpeedTracer**, we present some sample reports here. Three types of reports are generated by **SpeedTracer**: user reports, path reports, and group reports. These reports are generated in HTML format so that one can view the reports through a browser from the Internet or an intranet (see Figure 6). Hot links are also provided so that one can click on them through a browser and go to the original pages to see what they are. Java** applets are used to show various charts in the user reports.

To demonstrate some of the features of **SpeedTracer**, we processed the server log files generated at one of the IBM Web sites running IBM ICS. There were 37 984 entries in the access, referrer, and agent log. Note that the three IBM ICS logs contain exactly the same number of entries. However, there might be a big discrepancy among the three log files from other Web servers, such as the NCSA HTTPd. Sometimes log entries are simply dropped by the Web server in one of the files. Special logic in **SpeedTracer** is designed to synchronize the three files.

Figure 7 shows a snapshot of two statistics from the user reports. One is the top 10 most frequent external referrers to the server site, and the other is the top 10 most frequently visited pages. Note that "external" is with respect to the server site whose log files are being analyzed. The largest user count in the external referrer table is "no referrer." When a user visits a URL from his or her bookmark or by directly typing in the URL, there is no referrer information for such an access. But, the large count is due mostly to the fact that many of the accesses involve CGI (Common Gateway Interface) programs. As a result, these accesses were treated as single-- page sessions. The most frequent external referrer table can be used to measure the effectiveness of Web advertisements. It indicates the top external URLs from which most user sessions begin. If one has paid to place an advertisement on a certain site but the user count for this external referral is consistently low, one immediately realizes that the money was not well spent.

In the table for the most frequently visited pages in Figure 7, both click counts and user counts are provided. Differences between click and user counts do exist, and some of them can be substantial. As expected, the most frequently visited page by user count is the home page (/). However, on other experiments, we found that the home page is not always the hottest page visited by users because some users may have bookmarked some page other than the home page and go directly to it. In fact, this can be verified by the statistics that show the top pages to which users most often enter a Web site as provided by **SpeedTracer** (not shown here).



Enlarge 200%
Enlarge 400%

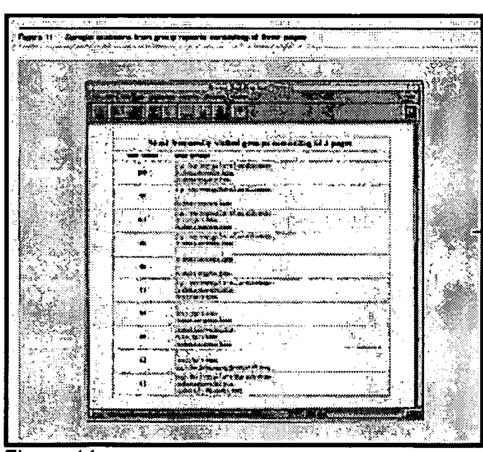
Figure 10

Figure 8 shows other interesting statistics from the user reports: the top 20 IP/host names from which most users come to visit the Web site, the total number of user sessions from each IP/host, the average duration, and the number of pages visited for these user sessions. There are a lot of single-page sessions because of the lack of referrer information. The overall distributions of user session duration and number of pages visited by users are also provided in Figure 9. The majority of the user sessions last less than 10 minutes and are visited by less than 10 pages. Java applets were used to draw charts on the user's browser based automatically on the data in the report.

Figure 10 shows sample statistics from the path reports. **SpeedTracer** presents the top N most frequently visited paths with different numbers of links. In Figure 10, we only present the top 10 most frequently visited paths with two links. These paths are forward paths, meaning that one can follow the path to visit each page. Figure 11, in contrast, shows the top 10 most frequently visited groups consisting of three pages. These pages may or may not be on the same path. Even if they are, they may be visited at various times via other intermediate pages. For example, the top path in Figure 10 and the top group in Figure 11 contain the same three pages. But their user counts are different. The user count for the path is less than that for the group of pages because there are many different ways to visit these three pages.

By comparing Figures 10 and 11, we notice that only the first group and the lOth group in Figure 11 are also among the top 10 frequented paths in Figure 10. However, seven of these eight remaining groups contain pages on the top 10 paths with five links (see Figure 12) except (/, /ics/icfgive.htm, and /cgibin/htimage/gifs/anim18.map). Such findings suggest that many users may have traveled very deep through various paths before they found the commonly desired pages. A simplified design to shorten the depth of the traversal paths might be warranted. Since HT=P links are typically embedded in a very complex way, an examination of both frequently visited paths and groups can help a Web site to better organize its presentation.

Summary

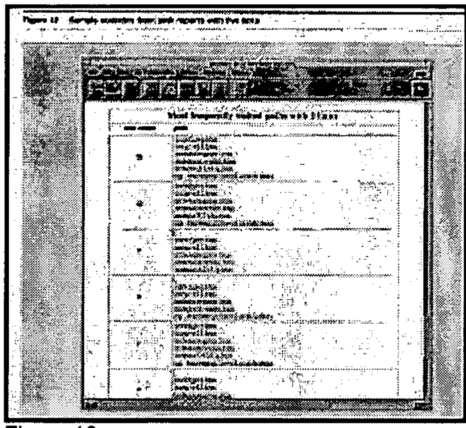


Enlarge 200%
Enlarge 400%

Figure 11

In this paper, we described the design of **SpeedTracer**, a Web usage mining and analysis tool. It reconstructs user traversal paths to identity user sessions even if user identities are hidden behind proxy servers or firewalls. No "cookies" or user registration are required for user session identification. User-- oriented statistics are provided, such as the most frequent external referrers, the most frequently visited pages, the distributions of user session durations and number of pages visited. In addition, the most frequented traversal paths and the most frequently visited group of pages are also reported by **SpeedTracer**. We also presented a few snapshots of sample reports generated with **SpeedTracer**.

Acknowledgment



Enlarge 200%

Enlarge 400%

Figure 12

The authors would like to express their sincere gratitude to Robert Kreigh, Michael Stokes, Arnold Goldberg, and Ajay Balusu at IBM Raleigh for their helpful discussions during the course of developing **SpeedTracer** as part of various IBM Lotus Go, Lotus Go Pro, and OS/390* ICss version 2.2 product offerings.

[Sidebar]

[Sidebar]
SpeedTracer, a World Wide Web usage mining and analysis tool, was developed to understand user surfing behavior by exploring the Web server log files with data mining techniques. As the popularity of the Web has exploded, there is a strong desire to understand user surfing behavior. However, it is difficult to perform user-- oriented data mining and analysis directly on the server log files because they tend to be ambiguous and incomplete. With innovative algorithms, **SpeedTracer** first identifies user sessions by reconstructing user traversal paths. It does not require "cookies" or user registration for session identification. User privacy is protected. Once user sessions are identified, data mining algorithms are then applied to discover the most common traversal paths and groups of pages frequently visited together. Important user browsing patterns are manifested through the frequent traversal paths and page groups, helping the understanding of user surfing behavior. Three types of reports are prepared: user-based reports, path-based reports and group-based reports. In this paper, we describe the design of **SpeedTracer** and demonstrate some of its features with a few sample reports.

[Footnote]
*Trademark or registered trademark of International Business Machines Corporation.
**Trademark or registered trademark of Microsoft Corporation, Bien Logic, Inc., Sane Solutions, LLC, Software, Inc., Netscape Communications Corp., or Sun Microsystems, Inc.

[Reference]
Cited references and notes

[Reference]
The uniform resource locator is used to uniquely identify a resource on the Internet. An example of a URL is "http: //www.ibm.com/," which represents the IBM home page on the Internet.
HyperText Transfer Protocol is the basic protocol used by the Web to transfer documents between a browser and a Web server.
3. J. Pitkow, "In Search of Reliable Usage Data on the WWW," Proceedings of Sixth International World Wide Web Conference (1997).
The National Center for Supercomputing Applications is located in the University of Illinois at Urbana-Champaign, Illinois.

[Reference]
5. NCSA HTTPd is an HTTP/1.0-compatible server for making hypertext and other documents available to Web browsers. It is copyrighted by the University of Illinois and owned by the university.
6. **SpeedTracer** is available for download from IBM Alpha Works(TM) at http://www.alphaworks.ibm.com. 7. B. Mobasher et al., Web Mining: Pattern Discovery from World Wide Web Transactions, Technical Report 96-050, Department of Computer Science, University of Minnesota, Minneapolis (September 1996).
8. P. Pirolli, R. Rao, and J. Pitkow, "Silk from a Sow's Ear: Extracting Usable Structures from the Web," Proceedings of 1996 Conference on Human Factors in Computing Systems (1996), pp. 118-125.
9. R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," Proceedings of ACM SIGMOD International Conference on Management of Data (1993), pp. 207-216.
10. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," Proceedings of Very Large Data Bases (1994), pp. 478-499.

[Reference]
11. J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," Proceedings of the 21 st VLDB Conference (1995), pp. 420-431.
12. J.-S. Park, M.-S. Chen, and P. S. Yu, "An Effective Hash Based Algorithm for Mining Association Rules," Proceedings of ACM SIGMOD International Conference on Management of Data (1995), pp. 175-186.
13. R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proceedings of 11th International Conference on Data Engineering (1995), pp. 3-14.
14. M.-S. Chen, J. S. Park, and P. S. Yu, "Data Mining for Path Traversal Patterns in a Web Environment," Proceedings of International Conference on Distributed Computing Systems (1996), pp. 385-392.
Accepted for publication August 5, 1997.

[Author Affiliation]
Kun-Lung Wu IBM Research Division, Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598 (electronic mail: klwu@watson.ibm.com). Dr. Wu received a B.S. degree in electrical engineering from the National Taiwan University in 1982 and M.S. and Ph.D. degrees in computer science from the University of Illinois at Urbana-Champaign in 1986 and 1990, respectively. From 1985 to 1989 he was a research assistant at the Center for Reliable and High-Performance Computing, Coordinated Science Laboratory, University of Illinois at Urbana-- Champaign. In the summer of 1986 he also worked as a consultant at Texas Instruments. Since March 1990 Dr. Wu has been with the IBM Watson Research Center, where he is currently a research staff member in the Software Tools and Techniques group in the Internet Technology Department. His current research interests include data mining tools for the World Wide Web, Internet applications, interactive information warehousing, database transaction and query processing, multimedia system designs, and network-centric information services. Dr. Wu has served as an organizing and program committee member for various IEEE conferences, and he is a member of the IEEE, ACM, and Phi Kappa Phi.

[Author Affiliation]
Philip S. Yu IBM Research Division, Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598 (electronic mail: psyu@watson.ibm.com). Dr. Yu received a B.S. degree in E.E. from the National Taiwan University in 1972, M.S. and Ph.D. degrees in E.E. from Stanford University in 1976 and 1978, respectively, and an M.B.A. from New York University in 1982. He has been with the IBM Watson Research Center since 1978 and is currently the manager of the Software Tools and Techniques group in the Internet Technology Department, of which one project focuses on developing algorithms and tools for Internet applications, such as the Web usage mining tool. His current research interests include data mining, Internet applications, database systems, multimedia systems, parallel and distributed processing, disk arrays, computer architecture, performance modeling, and workload analysis. Dr. Yu has published more than 210 papers and over 150 research reports and invention disclosures. He holds or has applied for 50 U.S. patents. He is a Fellow of the ACM and the IEEE and was an editor of IEEE Transactions on Knowledge and Data Engineering, also serving as a guest coeditor of a special issue on mining of databases. In addition to serving as program committee member of various conferences, he was the program cochair of the llth International Conference on Data Engineering and the program chair of the 2nd International Workshop on Research Issues on Data Engineering: Transaction and Query Processing. He will be the general chair of the

[Author Affiliation]
14th International Conference on Data Engineering. Dr. Yu has received several honors, including Best Paper Award, and from IBM two Outstanding Innovation Awards, an Outstanding Technical Achievement Award, a Research Division Award, and 19 Invention Achievement Awards.
Allen Ballman IBM Research Division, Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598. Mr. Ballman is a part-time researcher at the IBM Watson Research Center, exploring Web usage mining tools. Along with his interest in developing tools for the Internet, his research interests are concentrated in the areas of network-based simulation techniques, caching proxy servers, multimedia databases, and memory cache designs. Mr. Ballman received his B.A. in computer science and mathematics from Macalester College in 1992 and is currently pursuing a Ph.D. in computer science at the State University of New York-Stony Brook. He is a member of the ACM and USENIX.